RENAULT NISSAN

RENAULT NISSAN DESIGN SPECIFICATION (RNDS)

Part/module generic specifications, containing
RNDS-B-20137 v1.1

RENAULT STANDARD (GDNormes)	NISSAN DESIGN SPECIFICATION (NDS)
-	-

Issued: 2020-08-19

IMPORTANT PART SYMBOL

Title: Generic Security Specification

Confidentiality level: Alliance Internal

Signature (Alliance Single Technical Leader)

Stéphane BEROFF

Signature (Renault validator)

Robert Mathieu

Signature (Nissan validator)

Shiro Tsujioka

Revisions

Version	Date	Author	Revision History
1.1	2020-08-19	Cincilla, Pierpaolo Gentilhomme, Olivier F. Suzuki	Updated
1.0	2019-03-31	Philippe Quere S.Matsuyama	Newly established

Important Notices and Disclaimers

- It complies with the agreement reached between RENAULT and NISSAN in 2020-08-19.
  Any revision or alteration of this document is subject to prior approval by the RENAULT or NISSAN secretariat.
- The original version of this document was written in English.
  In the event of any discrepancies or differences in meaning created as a result of translation of this document into a foreign language, the meaning found in the English original shall take precedence.

Foreword

This document re-uses paragraph(s) of RENAULT document “Security Generic Specification”.
When the content of that paragraph(s) is(are) altered a notice shall be given to RENAULT secretariat

Renault	Issued by	:	Cincilla Pierpaolo (DEA-C) Gentilhomme Olivier (DEA-C)
	Validated by	:	Robert Mathieu (DEA-C)
NISSAN	Issued by	:	Fumihisa Suzuki (XN2)
	Validated by	:	Masaaki Miyashita (XN2)

Contents

Introduction

This document is the generic security requirements that can be applied for the attack surface ECU exposed to external interfaces such as “IVI, IVC, T-box, and BLM”. The actual application is specified by the OEM in the part specific requirements.

The requirements are sorted in below categories:

1. Firmware
2. Private Keys management
3. User profiles
4. IP Connectivity (OSI 1, 2, 3, 4): (MNO, Bluetooth (PAN), Wi-Fi, ...)
5. Connectivity (others: Bluetooth audio, USB mass storage, SD card, ...)
6. Applications
7. Internet Browser
8. Off-board services (OSI 5, 6, 7)
9. CAN controller
10. Development / Serial Life

Generic Security Specification

1 Scope

This document is applicable for Cybersecurity related parts.
The actual application is specified by the OEM in the part specific requirements.

2 Normative references

These normative and informative references apply the latest version.
[REF 1] BMIR-V5636-2007-0012-V11 Template_SOW

3 Terms and definitions

For the purpose of this standard, the following terms and definitions apply.

The following terminology is used to define the applicability of each requirement in this document.

- The word [PLATFORM] in the text is defined by each cybersecurity related part.
- The word **SHALL** in the text means a mandatory requirement.
- The word **SHOULD** in the text means a recommendation or advice on implementing a requirement. Such recommendations or advice are expected to be followed; unless justified reasons are stated for not doing so.

Requirements are noted as follows:

Req_Id Sub_Req_Id	**Req_Text**	**Req_Flexibility**

Req_Id: is the unique requirement identifier for the whole project.
Sub_Req_Id: when present, is the unique requirement identifier for the whole project and in this case **Req_Id** becomes the aggregation of all related Sub_Req_Id.
Req_Text: is a description of the requirement.
Req_Flexibility: is the requirement flexibility level assigned to the supplier:

- **0** = no flexibility, requirement has to be met
- **1** = low flexibility, requirement barely negotiable
- **2** = high flexibility, negotiable requirement

USER INTERFACE: A user interface is an interface on which a user controllable flow could be pushed. It's not HMI; it's really the physical interface on which the user can provide some inputs.
Examples for:

- Multimedia: User interfaces are USB, SD Card, mass storage …
- TCU: User interface is Serial over USB
- Multimedia embedded CAN Controller: User interface is the link (UART …) between the main SOC and the CAN controller

REMOTE INTERFACE: Are the same of user interface but user controllable flows are authenticated by a remote provider (e.g. MNO for 2G/3G access, content aggregator for Store …).

Examples for:

- Multimedia: Remote interfaces are TCU, Bluetooth PAN, Wi-Fi, IP over USB
- TCU: Remote interfaces are 2G/3G
- Multimedia embedded CAN Controller: FOTA function

OEM INTERFACE: Are only accessible by physical access on board (JTAG, UART in certain circumstances, and by extension CAN because it is supposed to be only accessible by car manufacturer).

Examples for:

- Multimedia: JTAG, UART for debug board on previous multimedia version
- TCU: JTAG, CAN-V, CAN-M
- Multimedia embedded CAN Controller: CAN-V, CAN-M, JTAG

4 Symbols and abbreviations

For the purpose of this standard, the following symbols and abbreviated terms apply.

Acronym/Other	Description
SDK	Software Development Kit
HMI	Human Machine Interface
TCU	Telematics Control Unit
CAN	Controller Area Network
UART	Universal Asynchronous Receiver/Transmitter
JTAG	Joint Test Action Group
TPM	Trusted Platform Module
SRK	Strategy Root Key
PKI	Public Key Infrastructure
RSA	Rivest-Shamir-Adleman cryptosystem
DSA	Digital Signature Algorithm
ECDSA	Elliptic Curve Digital Signature Algorithm
AES	Advanced Encryption Standard
SHA	Secure Hash Algorithm
PBKDF2	Password-Based Key Derivation Function 2
RFC	Request For Comments
DES	Data Encryption Standard

ANSSI	Agence nationale de la sécurité des systèmes d'information
NIST	National Institute of Standards and Technology
OCSP	Online Certificate Status Protocol
ADB	Android Debug Bridge
MMU	Memory Management Unit
HSM	Hardware Security Module
CSR	Certificate Singing Request
GDM	GNOME Display Manager
KDM	KDE Display Manager
PAN	Personal Area Network
MNO	Mobile Network Operator
VPN	Virtual Private Network
VIN	Vehicle Identification Number
PIN	Personal Identification Number
Android Dalvik VM	Android Dalvik Virtual Machine
Android JNI calls	Android Java Native Interface calls
SDLC	Secure Development Life-Cycle
IPSec (with IKE v2)	Security Architecture for Internet Protocol
IKE v2	Internet Key Exchange
SSH v2	Secure Shell
EAL4+	Evaluation Assurance Level
FIPS	Federal Information Processing Standards

5 Firmware

This section describes the software running on the [PLATFORM]. The goal is to provide a consistent subset of requirements to manage boot, upgrade and run-time of the [PLATFORM] and it also covers basics for data protection (additional requirements will be found in each necessary following section)
This section does not cover anything about services, applications or communications. For these topics, refer to the dedicated section.

5.1 Secure boot

FRQ_FMW_SBO_01x	**Secure Boot**	
FRQ_FMW_SBO_011	The system SHALL provide a mechanism to protect [PLATFORM] integrity during boot time.	**0**
FRQ_FMW_SBO_012	During each boot step, the [PLATFORM] SHALL verify the integrity of the code to be loaded before being executed. Therefore, integrity of OS and all SW components are preserved.	**0**
FRQ_FMW_SBO_013	Integrity check SHALL be enforced by robust cryptographic algorithms (see **FRQ_FMW_CRY_010** for asymmetrical cryptographic algorithms authorized) Example of such mechanism is what can be done with a TPM (Trusted Platform Module, http://www.trustedcomputinggroup.org/). Remark: This requirement SHALL apply to ALL systems loaded by the [PLATFORM] (firmware updater, hibernation resumption, etc …)	**0**
FRQ_FMW_SBO_014	Platform integrity SHALL be checked in nominal case after each switch on of the vehicle or after more than 24 hours between two switches on.	**0**
FRQ_FMW_SBO_02x	**Secure Boot keys management**	
FRQ_FMW_SBO_020	Each boot step SHALL use different asymmetrical key pairs (The objective is to enforce different security policy according to frequency usage of these key pairs).	**0**

FRQ_FMW_SBO_021	Different parts (for example development and serial life parts) SHALL use different key pairs, see **FRQ_DSL_HWR_011 and FRQ_DSL_HWR_024**	**0**
FRQ_FMW_SBO_025	**Secure Boot public keys revocation** The [PLATFORM] SHALL provide a mean to revoke the hardware implemented public key.	**1**
FRQ_FMW_SBO_026	The [PLATFORM] SHALL provide a mean to revoke the Bootloader public key.	**1**
FRQ_FMW_SBO_027	The [PLATFORM] SHALL provide a mean to revoke the Kernel public key. Examples: - Hardware implemented public key : SRK for TrustZone Freescale's implementation - Bootloader and Kernel keys COULD be revoked through firmware upgrade	**1**
FRQ_FMW_SBO_03x	**Secure Boot – Specific implementation**	
FRQ_FMW_SBO_030	In case of the main operating system image is too big to verify at boot time, it could be convenient to enforce signature verification at run-time This could be done only if the kernel below has the ability to run the verification code at the time it loads the userland application in memory and just before execution. This could be achieved with kernel hooks (it implies that kernel is open-source or at least offers hooks for these specific needs) Example: • See dm-verity for Android 4.4 (https://source.android.com/security/verifiedboot/dm-verity)	**1**

FRQ_FMW_SBO_04x	**Platform integrity**	
FRQ_FMV_SBO_041	In continuity of the Secure Boot, the [PLATFORM] SHALL authenticate every executable code which is to be run.	**0**
FRQ_FMV_SBO_042	If authentication could not be enforced, then the code SHALL NOT be executed on the [PLATFORM]. Example: - Any user application installed through the Store - Any executable on removable devices (USB, SD Card)	**0**

5.2 Firmware Update

Reactivity and reliability of the updating process is a key aspect.

Delivery of patch against security breaches should be made available to the [PLATFORM] units in short-time (e.g. delay in the range of a few days).

NOTE: A specific contractual commitment SHALL be made between Renault-Nissan and Supplier for the maintenance. Maintenance includes security fixes, corrective, improvement, preventive, or adaptive to new environment. This is specified in the Statement Of Work (SOW) document of the RFQ package. See "§ 6.10.6 CYBERSECURITY MAINTENANCE", requirements [SECU-700] and [SECU-710].

FRQ_FMW_UPD_01x	**Firmware update**	
FRQ_FMW_UPD_010	A firmware update function SHALL be implemented on the [PLATFORM] through at least one of the available interfaces: 1- User interface: USB, SD Card, customer's smartphone as mass storage …) 2- Remote interface: OTA through any network interface (Wi-Fi, 2G, 3G, Bluetooth, customer's smartphone as network bridge, …) 3- OEM interface: CAN, JTAG, UART … Remark: See **§ Definitions** for User interface, Remote interface and OEM interface definitions	**0**

FRQ_FMW_UPD_02x	**Firmware update image signature**	
FRQ_FMW_UPD_021	Updater SHALL verify and authenticate firmware update image by using robust cryptographic algorithms (see **FRQ_FMW_CRY_010** for asymmetrical cryptographic algorithms authorized).	**0**
FRQ_FMW_UPD_022	Key pair used to sign firmware update image SHALL be distinct from those used in other functions like Secure Boot.	**0**
FRQ_FMW_UPD_023	Key pair used to sign Secure software and unsecure software SHALL be different.	**0**
FRQ_FMW_UPD_024	To perform the verification of the signature, control means SHALL be embedded in the source code of the firmware already installed on the ECU. *The control means* SHALL *not be delivered with the firmware update* Control means : • If the private signature key has a public key not certified, the public key SHALL be embedded in the firmware already installed on the ECU. This public key SHALL NOT be sent with the firmware update • If the private signature key has his public key signed by a PKI (certificate), the firmware already installed SHALL embed the trust chain or Root CA certificate. In this case, the signature certificate SHALL be sent with the firmware update. Note : the public key SHALL NOT be auto-certified.	**0**
FRQ_FMW_UPD_03x	**Firmware update image encryption**	
FRQ_FMW_UPD_031	Firmware update image shall be encrypted by robust cryptographic algorithms (see **FRQ_FMW_CRY_010** for symmetrical and asymmetrical cryptographic algorithms authorized) to protect software from disclosure.	**0**

FRQ_FMW_UPD_032	Key used for firmware update image encryption SHALL be distinct from those used in other functions.	**0**
FRQ_FMW_UPD_033	Storage of the encryption key SHALL follow req **FRQ_FMW_STO_03x.**	**0**
FRQ_FMW_UPD_04x	**Firmware versioning**	
FRQ_FMW_UPD_040	Firmware images SHALL include a version number.	**0**
FRQ_FMW_UPD_05x	**Firmware downgrade**	
FRQ_FMW_UPD_050	Updating [PLATFORM] with a strictly lesser version number SHALL be forbidden. Updating with a greater or equal version number is allowed.	**0**

5.3 Cryptography

FRQ_FMW_CRY_01x	**Cryptography functions**	
FRQ_FMW_CRY_010	The [PLATFORM] implementation SHALL provide support for cryptographic functions required for secure communication and for other possible security services (e.g. user identification/authentication…). This includes the cryptographic support for (*): - **Asymmetric**: • RSA (2048 bits minimum) • DSA (L=3072 bits and N=256 bits minimum) • ECDSA (between 256 and 383) - **Symmetric**: • AES (128 bits minimum) • Blowfish (128 bits minimum and 16 rounds minimum) • Twofish - **Hash-functions**: • SHA-2 • SHA-3 - **Key derivation function**:	**0**

	• PBKDF2 (Salt : 128 bits minimum, 1000 iterations minimum, see Hash-functions authorized) **- Randomness** • The generation of random number for cryptographic operations SHALL follow RFC 4086 According to ANSSI recommendations: • MD5 and SHA-1 as hash functions SHALL be avoided • 3DES as symmetric encryption SHOULD be avoided This requirement follows NIST recommendation : SP800-57 and SP800-132 (*) at least support one algorithm for each family	
FRQ_FMW_CRY_02x	**Cryptographic services**	
FRQ_FMW_CRY_020	The [PLATFORM] system SHOULD provide an API for cryptographic functions described in **FRQ_FMW_CRY_010**. Example: Applications rely on this API for all cryptographic operations.	**1**

5.4 TLS

FRQ_FMW_TLS_0xx	**General**	
FRQ_FMW_TLS_001	All connections to the Alliance off-board platform SHALL be secured using TLS and mutually authenticated using the client certificate provisioned in the [PLATFORM].	**0**
FRQ_FMW_TLS_002	TLS client SHALL close TLS session in case of: • Certificate Status equal to Unknown • Certificate Status equal to Revoked • Any Exception cases	**0**
FRQ_FMW_TLS_003	TLS Server certificate should include OCSP server URL. The TLS client SHALL not manage URL missing as an issue.	**1**
FRQ_FMW_TLS_004	The TLS client SHALL support the "Server Name" extension (SNI) as described in RFC 6066. ServerName provided in TLS extension will be the same value than the one present in the URL.	**0**
FRQ_FMW_TLS_005	TLS implementation SHALL only support protocol version 1.2	**0**

	and superior versions. The latest version SHALL be first used (as much as possible).	

FRQ_FMW_TLS_1xx	**TLS 1.2**	
FRQ_FMW_TLS_10x	**TLS Client**	
FRQ_FMW_TLS_101	Client-side initiated renegotiation SHALL be disabled	**0**
FRQ_FMW_TLS_102	Insecure renegotiation SHALL be disabled	**0**
FRQ_FMW_TLS_103	Cipher suites containing at least one of the following option SHALL be forbidden: • Null cipher; • RC4 cipher; • 3DES cipher; • Export key exchange; • Anonymous Diffie Hellman	**0**
FRQ_FMW_TLS_104	The following cipher suites SHALL be preferred when available: • TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 • TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 • TLS_DHE_RSA_WITH_CHACHA20_POLY1305_SHA256 • TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 • TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 • TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 • TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 • TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 • TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	**0**
FRQ_FMW_TLS_105	The following ciphers suites are allowed as a fallback when preferred cipher suites are unavailable: • TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 • TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 • TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 • TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 • TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 • TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 • TLS_ECDHE_RSA_WITH_CAMELLIA_256_CBC_SHA384	**0**

	• TLS_ECDHE_RSA_WITH_CAMELLIA_128_CBC_SHA256 • TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA256	
FRQ_FMW_TLS_106	The TLS client SHALL be able to manage session tickets (as per RFC 5077). Session ticket information might be stored in a non-volatile memory.	**0**
FRQ_FMW_TLS_107	The TLS client SHALL support and use first and foremost the "Certificate Status Request List" (aka "Multi OCSP Stapling") TLS extension following the RFC 6961.	**1**
FRQ_FMW_TLS_108	The TLS client SHALL support the "Certificate Status Request" (aka "OCSP Stapling") TLS extension following the RFC 6066. This extension SHALL be used if the Multi OCSP Stapling is not supported.	**0**
FRQ_FMW_TLS_109	OCSP Stapling SHALL follow FRQ_FMW_CRY_010	**0**
FRW_FMW_TLS_110	The DHE-based key exchange is authorized only if the order is at least 2048 bits.	**0**

FRQ_FMW_TLS_2xx	**TLS 1.3**	
FRQ_FMW_TLS_20x	**TLS Client**	
FRQ_FMW_TLS_201	The TLS client SHALL only use the mode "PSK with (EC)DHE" when resuming a TLS session using a ticket (PSK). Note: PSK-only mode is prohibited.	**0**
FRQ_FMW_TLS_202	The TLS client shall never use a ticket more than once.	**0**
FRQ_FMW_TLS_203	The TLS client SHALL not use 0-RTT unless it is specifically requested by the application.	**0**
FRQ_FMW_TLS_204	If the early data are rejected by the TLS server, thus the TLS client SHALL not automatically resend those data as applicative data (once then handshake is done) unless it is instructed by the application.	**0**
FRQ_FMW_TLS_205	The TLS client shall provide a way for the application to determine if the handshake has completed.	**0**
FRQ_FMW_TLS_206	The TLS client shall provide a way for the application to determine whether the early data have been accepted.	**0**
FRQ_FMW_TLS_207	The TLS client shall always add the "status_request" extension in its "clientHello" message and shall always check **all** the OCSP responses (covering at least the TLS server certificate) sent in the "ServerHello" message. The connection shall also be closed by the client whether an OCSP response is not "OK" or if its signature cannot be validated.	**0**
FRQ_FMW_TLS_21x	**Client application behavior (MQTT client, HTTP client, …)**	
FRQ_FMW_TLS_210	The applicative client shall not use TLS with 0-RTT without defining a security policy to use it (a set of requests/data safe	**1**

	to be replayed).	
FRQ_FMW_TLS_211	If the applicative client is not able to define such a security policy for 0-RTT, it shall either not use 0-RTT at all or implement a smart retry strategy like {try, poll (wait-read), retry} in case the 0-RTT failed (no response or early data refused).	**0**
FRQ_FMW_TLS_22x	**TLS Server (final or proxy)**	
FRQ_FMW_TLS_220	The TLS server may emit multiple tickets after a successful handshake.	**0**
FRQ_FMW_TLS_221	The TLS server SHALL refuse any session resumption based on a "PSK-only" mode.	**0**
FRQ_FMW_TLS_222	The TLS server SHALL only send a key identifier as a ticket to the TLS client. Note: Combined with a single ticket use strategy, it allows to enhance the forward secrecy for 0-RTT data.	**0**
FRQ_FMW_TLS_223	The TLS server SHALL implement a "single use ticket" anti-replay mechanism based on a consistent storage. The tickets database SHALL be synchronized between the different nodes.	**0**
FRQ_FMW_TLS_23x	**HTTP inspection by TLS servers (final or proxy)**	
FRQ_FMW_TLS_230	A TLS server (final or proxy) SHALL accept (or forward) HTTP-typed early data (0-RTT) if ONLY the following HTTP methods are in use: - HEAD, - OPTIONS, - GET without parameters.	**0**
FRQ_FMW_TLS_231	Upon detection/acceptance of HTTP early data, the TLS server (final or proxy) SHALL add the HTTP header "Cf-0rtt-unique: <*hash*>" if this latter is not already present. Note: The "<*hash*>" SHALL be a unique value derived from the PSK binder of the "ClientHello" message (which is a HMAC computed over a part of the ClientHello using the PSK). This value SHALL be derived using a robust hash algorithm (see **FRQ_FMW_CRY_010**).	**1**
FRQ_FMW_TLS_232	Upon detection/acceptance of HTTP early data, the TLS server (final or proxy) SHALL add the HTTP header "Early-Data: 1" if this latter is not already present (RFC 8470).	**0**
FRQ_FMW_TLS_24x	**HTTP servers behavior**	
FRQ_FMW_TLS_240	The HTTP server shall recognize the following HTTP headers as legitimate: "Cf-0rtt-unique" and "Early-Data".	**0**
FRQ_FMW_TLS_241	The HTTP server shall respond with the error code 425 ("Too early data") in the following cases: - It considers the data as not safe to be replayed,	**1**

	- It detects a 0-RTT replay (using "Cf-0rtt-unique" header).	
FRQ_FMW_TLS_242	The HTTP server shall accept the HTTP request if it is not able to act consistently (according to a security policy) when receiving the aforementioned headers.	**0**

5.5 Time Synchronization with Trusted Time source

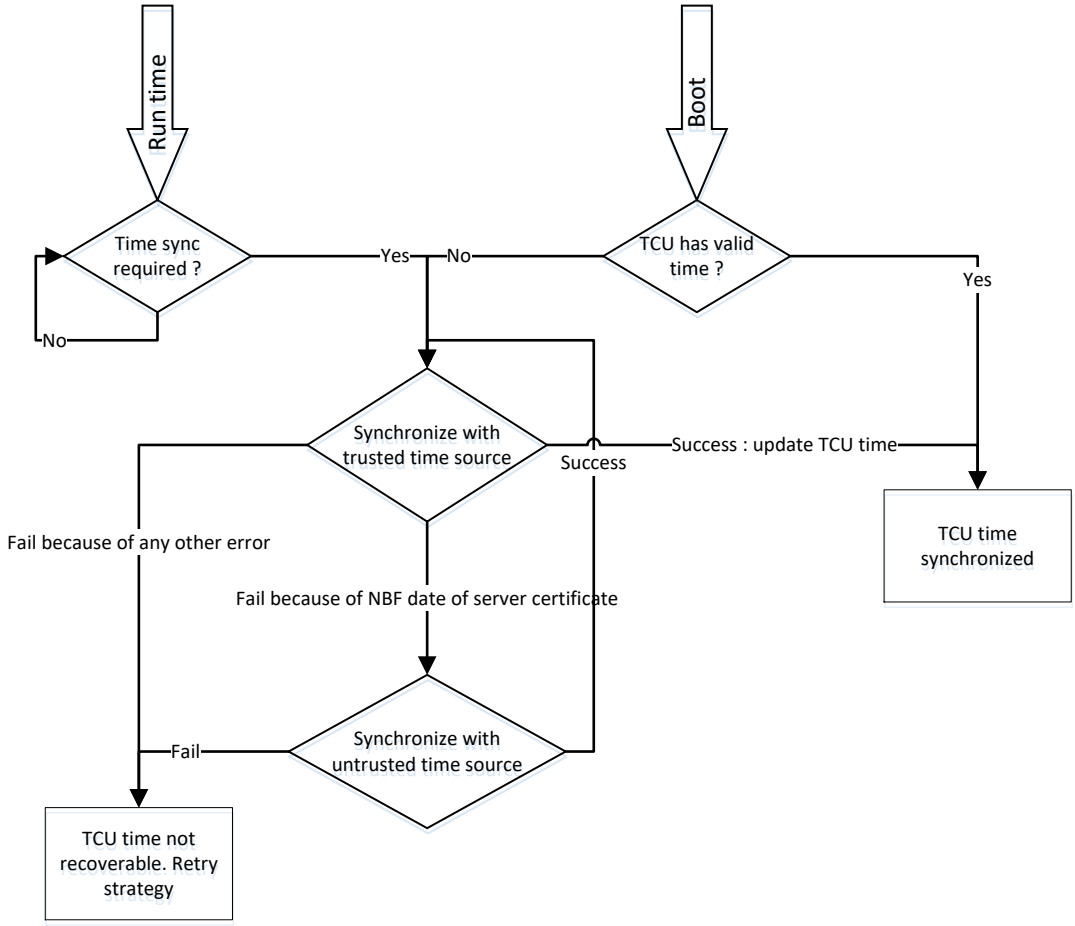
FRQ_FMW_TRT_01x	**Trusted-Time Server**	
FRQ_FMW_TRT_011	The [PLATFORM] System SHALL be able to synchronize its time with a Trusted Time Source.	**0**
FRQ_FMW_TRT_012	The [PLATFORM] System SHALL be able to synchronize its time source on a regular basis.	**0**
FRQ_FMW_TRT_013	The periodicity of this process SHOULD be configurable.	**1**
FRQ_FMW_TRT_014	The [PLATFORM] System SHALL use a Trusted Time Source while executing Security Operations.	**0**
FRQ_FMW_TRT_015	If Time Synchronization fails, [PLATFORM] System Time SHALL be considered as INVALID The [PLATFORM] System SHALL be able to evaluate if its time source IS VALID. For example: • Detection of backup battery removal shall induce an INVALID time. • Time earlier than 01/01/2019 shall be detected as INVALID • Time Synchronization is overdue	**0**
FRQ_FMW_TRT_016	If [PLATFORM] Time source is INVALID, then Trusted Time Source Synchronization SHALL be executed	**0**
FRQ_FMW_TRT_017	**Trusted Time Synchronization Method** Trusted Time Synchronization SHALL be based on “HTTP HEAD over TLS” algorithm. *See description of the solution below*	**0**

FRQ_FMW_TRT_018	**Exception: Synchronizing with untrusted sources.** If a synchronization with a Trusted Time Source is not possible (Trusted Time Server Certificate validation fails with "Not Valid Before" Error), then the [PLATFORM] Time source SHALL be synchronized with an **"untrusted source***", i.e. GPS Clock or Standard HTTP server.* **THEN subsequently**, it SHALL Synchronize its time source with a Trusted Time Source.	**0**

Trusted Time Source Synchronization Algorithm:

This solution has been designed to answer Trusted Time Server Requirements. It is based on "HTTP HEAD" requests.

The following flowchart describes the solution:



- Synchronizing with a Trusted Time Source

The [PLATFORM] sends an **HTTP HEAD** request to the trusted time **URL**. Trusted time **URL** is <*TBD*>. The trusted time URL sends back an empty response containing a Date header. The Date header is parsed by the [PLATFORM] and the local [PLATFORM] time

is set to this timestamp. This timestamp is also saved internally in non-volatile memory as the last saved synchronized time.

Example:

1. Establish a TLS connection to the trusted time url.
2. Send an HTTP head request

HEAD / HTTP/1.1
Host: <Trusted time url>
User-agent: TCU

3. Receive the HTTP response

HTTP/1.1 200 OK
Date: Wed, 08 Feb 2017 10:41:28 GMT

4. Parse the Date header
5. Set the [PLATFORM] local time
6. Save the last saved synchronized time.

- **Synchronize with untrusted time source**

The [PLATFORM] sends an HTTP HEAD request to the untrusted time URL. Untrusted time URL is <*TBD*>. The untrusted time URL sends back an empty response containing a Date header. The Date header is parsed by the [PLATFORM] and stored in a temporary variable. This variable is then used to validate the certificate in the following step: **"Synchronize with trusted time source"**.

Example:

1. Send an HTTP head request to the untrusted time url.

HEAD / HTTP/1.1
Host: <Untrusted time url>
User-agent: TCU

2. Receive the HTTP response

HTTP/1.1 200 OK
Date: Wed, 08 Feb 2017 10:41:28 GMT

3. Parse the Date header
4. Store the timestamp in a variable
5. Execute **"synchronize with trusted time source"** using the variable defined above as the current time instead of the local TCU time.

5.6 Data protection

FRQ_FMW_DAT_01x	**Restriction on Firmware image content**	
FRQ_FMW_DAT_011	Firmware image SHALL not contain any license file or authentication credentials	**0**
FRQ_FMW_DAT_012	These items SHOULD be provisioned during part customization, or during first usage, or by any other side channel.	**1**
FRQ_FMW_DAT_02x	**Debug mode**	
FRQ_FMW_DAT_020	The [PLATFORM] SHALL NOT contain any debug features after development stage or any user accessible diagnostic functions (See **Chapter 13 Development / Serial Life** for specifications related to development phase). Example: • UART • JTAG • ADB	**0**
FRQ_FMW_DAT_021	If the developer debugging interface is available, the [PLATFORM] SHALL control it and prohibit unauthorized access to it.	**0**
FRQ_FMW_DAT_03x	**Password protection**	
FRQ_FMW_DAT_030	The [PLATFORM] SHALL protect the confidentiality of the password (for all passwords) when it is typed. But depending on the type of the input (touchscreen, rotary button or keyboard) the visual feedback can be different.	**0**

5.7 System permissions

FRQ_FMW_PER_01x	**System permissions**	
FRQ_FMW_PER_010	The [PLATFORM] SHALL implement a security policy allowing or forbidding a process from connecting to any	**0**

	equipment (phone, 3G USB device, USB device, music player, internet, phone call, CAN-M, CAN-Ext, ECUs …). Example: The process in charge of the TLS communication SHALL not be able access the CAN buses.	
FRQ_FMW_PER_02x	**Secure execution environment**	
FRQ_FMW_PER_021	The [PLATFORM] SHALL protect the execution of each process.	**0**
FRQ_FMW_PER_022	Each process SHALL run in a dedicated memory area in which no other process can read or write. Examples: MMU (Memory Management Unit) on all modern processor. Kernel => direct access to physical memory Userland => access to virtual memory Application SandBox	**0**
FRQ_FMW_PER_03x	**File system permissions**	
FRQ_FMW_PER_030	The [PLATFORM] SHOULD provide a mean to segregate the access of file system based on process privileges. It should follow the principle of “least privileges” (what is not needed is forbidden). **FRQ_FMW_PER_010** is for run-time resources **FRQ_FMW_PER_030** is for file system resources (like file configuration, local database, …) Example: The process in charge of TLS Communication SHALL not be able access the CAN-V configuration.	**1**
FRQ_FMW_PER_04x	**Critical function isolation**	
FRQ_FMW_PER_041	The [PLATFORM] SHALL manage a specific isolation mechanism to prevent the misusage of critical function. Any function with eFuse / OTP manipulation (like requested on req **FRQ_FMW_SBO_025**) is considered as a critical function.	**0**

FRQ_FMW_PER_042	This isolation MAY be used for other security related function. Example: - Trust Zone - Virtualization	**2**

5.8 Resources allocation protection

FRQ_FMW_RES_01x	**Quota usage**	
FRQ_FMW_RES_010	The [PLATFORM] system will be running multiple applications at the same time. The system SHALL enforce quotas per application for different hardware resources: • CPU, • RAM, • flash memory or hard drive, • Network bandwidth • …	**0**
FRQ_FMW_RES_02x	**Quota reaction**	
FRQ_FMW_RES_020	Quotas failures SHOULD be handled by functional requirements. Examples: • RAM over 80% => kill process • CPU over 90% during 50 seconds => reboot • Modifying CPU scheduling priorities	**2**

5.9 System reset / wipe

This will guarantee the user that if he sells his car; the next driver will not have access to his personal information. This is also true for rental companies. Between two users, the rental company performs a reset to be sure that no trace of the user is kept on the [PLATFORM]

The actions to perform during reset are:

- Erase all personal data
- Erase all profiles

- Erase Bluetooth pairing

Confidentiality levels:

- **C1:** critical. Personal data from users: names, emails, phone numbers, credit cards, GPS history …
- **C2:** standard. Device identification data: serial number, versions, battery state …
- **C3:** minimum. Public data: RSS feeds, internet content, audio streaming …

FRQ_FMW_WIP_01x	**Reset all data**	
FRQ_FMW_WIP_010	[PLATFORM] SHALL return to factory state with current firmware version after the system reset is requested by the user (e.g. before changing of car owner/ driver or car end-of-life). The factory state is the nominal status of the system (e.g. no add-on apps, no users configurations, no personal data) for the current running firmware version.	**0**
FRQ_FMW_WIP_02x	**Wipe levels**	
FRQ_FMW_WIP_021	For **C1** data the [PLATFORM] SHOULD enforce a secure wipe. It means that related memory sectors or pages SHALL be rewritten at least once.	**1**
FRQ_FMW_WIP_022	For **C2** and **C3** data the [PLATFORM] SHOULD at least unlink the related files. It COULD also use secure wipe.	**2**

5.10 Secure storage

There is no ideal solution to protect information in a secure storage from the system itself (as the system SHALL access these information and as it could be compromised).
But we can enforce security measures to prevent other risks such as:

- Unprivileged code running on the platform
  - Android permissions framework
  - OpenAT userland and kernel land segregation
- Physical attacks
  - HSM
  - Smart Card

FRQ_FMW_STO_01x	**Privacy protection**	
FRQ_FMW_STO_010	To ensure privacy, cryptography SHALL be used to protect the stored data. The user COULD choose a password that will be used to derive an encryption key in order to encrypt the data related to his profile in the [PLATFORM] and in the remote servers. This password COULD also be the one coupled to his profile. Example: Typically, it could be done through PIN code derivation and used for storing certificates or application credentials	**0**
FRQ_FMW_STO_02x	**Storage of non repudiable, authenticated data (audit logfiles)**	
FRQ_FMW_STO_020	Cryptographic functions SHOULD be used to store non-repudiable, auditable information which are stored in the [PLATFORM] in an authenticated/digitally signed way for further auditing process. Example**:** Storing of some purchasing or billing information in case of paying multimedia services, energy consumption in smart charging… It could be done by: • TPM: TPM stores hash of log files to ensure their integrity • Asymmetrical algorithms (private key stored locally) • Symmetrical algorithms	**1**
FRQ_FMW_STO_03x	**Device data protection**	
FRQ_FMW_STO_031	A dedicated permission (as described in FRQ_FMW_PER_010) SHALL protect device specific data (device's certificate, off-board authentication credentials, APN credentials, SMS key …)	**0**
FRQ_FMW_STO_033	The [PLATFORM] SHALL have a hardware vault like HSM (Hardware Security Module). This vault physically protects the secret keys used to encrypt confidential data on the system. It also allows to encrypt / decrypt the data for	**0**

	system use without the system actually knowing the encryption key. Example: SNVS of IMx6: Secured Non Volative Storage with OTPMK differentiation.	
FRQ_FMW_STO_034	**This hardware vault SHALL be used for Private Keys Management (see requirements FRQ_PRIV_KEY_xxx).**	**2**

5.11 Tracability

FRQ_FMW_TRA_04x	**Logs**	
FRQ_FMW_TRA_041	OS SHALL have the function of logging security events and SHALL support the audit function.	**0**
FRQ_FMW_TRA_042	An access control mechanism SHALL be adopted to log files to manage the read and write permissions of them.	**0**
FRQ_FMW_TRA_043	The log storage SHOULD be protected and stored in a secure environment.	**1**
FRQ_FMW_TRA_044	Personal sensitive information, including biometric information, SHALL NOT be written in log or SHALL be written after anonymization.	**0**
FRQ_FMW_TRA_045	Log data SHALL be kept for a sufficient period.	**0**
FRQ_FMW_TRA_046	C1/C2 on-board services SHALL log all accesses to C1/C2 data. The access logs SHALL have columns which can identify following information. -who accessed the information -when information was accessed *Access log to Cx data has the Cx confidentiality and SHALL comply with associated requirements.	**0**
FRQ_FMW_TRA_047	All logs SHALL be timestamped. The logs to be output are, for example, as follows. Example: - Authentication(Login) log: Authentication success/fail, Account change/create/delete, Use privileges - Communication log: Client request, Server response, IP address, protocol, etc - Error log - etc	**0**

6 Private keys management

This section describes the requirements needed in private keys management used by off-board systems for software signing (firmware, apps, …). Any use of asymmetrical cryptography relies not only on algorithm strengths and keys size but also on private key protection and generation.
It is completely useless to deploy asymmetrical cryptography without taking care on private keys.

FRQ_PRIV_KEY_01x	**Private keys management**	
FRQ_PRIV_KEY_010	All asymmetrical keys used for authenticate binary running on the [PLATFORM] SHALL be released and managed by Renault-Nissan.	**0**
FRQ_PRIV_KEY_012	A signature process SHALL be established and contracted between Renault-Nissan and Supplier.	**0**
FRQ_PRIV_KEY_02x	**Private key generation**	
FRQ_PRIV_KEY_021	The [PLATFORM] SHALL be able to generate a private key (according to **FRQ_FMW_CRY_010**).	**0**
FRQ_PRIV_KEY_022	The [PLATFORM] SHALL be able to build a Certificate Signing Request (CSR) based on the previous private key.	**0**
FRQ_PRIV_KEY_023	The [PLATFORM] CSR SHALL be accessible via **OEM** interface (diagnostic request over CAN).	**0**
FRQ_PRIV_KEY_024	The [PLATFORM] SHALL retrieve the signed certificate through **USER** or **REMOTE** interface before initiating any functional services.	**0**
FRQ_PRIV_KEY_025	The [PLATFORM] SHALL validate that the signed certificate corresponds to the CSR initially generated.	**0**
FRQ_PRIV_KEY_026	The [PLATFORM] SHALL store the final signed certificate according to **FRQ_FMW_STO_03x** requirements.	**0**

7 User profiles

This section describes the requirements needed in user management and isolation. It also provides the mandatory user data protection.

7.1 User segmentation

FRQ_USR_SEG_01x	**User profile management**	
FRQ_USR_SEG_011	The [PLATFORM] SHALL ask user's authentication to manage profiles. Note: Profile management can be done both locally and via an online network.	**0**
FRQ_USR_SEG_012	Each profile SHOULD be related to a person	**1**
FRQ_USR_SEG_013	Each profile SHALL only use his personal information.	**0**
FRQ_USR_SEG_014	When a user wants to select a profile, a method SHALL authenticate him.	**0**
FRQ_USR_SEG_015	A guest account without authentication COULD be available.	**2**
FRQ_USR_SEG_016	A guest account without authentication SHALL NOT contain any personal data.	**0**
FRQ_USR_SEG_017	Passwords SHALL have high complexity (at least including numbers, uppercase and lowercase letters, and no less than 8 digits in length), and if an user try to set low-complexity password, it SHALL give a clear risk warning.	**0**
FRQ_USR_SEG_018	The OS SHALL make the following settings related to the user accounts and privileges; a) Useless accounts SHALL be disabled or deleted b) Login with Root user account SHALL be prohibited. C) Privilege escalation SHALL be restricted.	**0**
FRQ_USR_SEG_019	When an user input sensitive information to the [PLATFORM], [PLATFORM] SHALL ensure no other application can steal it. For example, use a secure soft keyboard tested by a third-party professional organization.	**0**
FRQ_USR_SEG_02x	**User profile selection**	

FRQ_USR_SEG_020	The profile management COULD be done like on Linux (like GDM or KDM) or Windows (winlogon): a login screen with the profile to choose and the related password to type in. Two operation modes can be used: • Identification: the user chooses which profile he wants to use, without typing the password. When an application needs the user's authentication, then a secure (no other processes can access the entered data) screen pops up asking for him to type the password. • Authentication: the user chooses which profile he wants to use and types in the password to be authenticated. By this mean, he will never have to type his password again to launch any application. In this case, the [PLATFORM] SHOULD provide a guest account to allow using the [PLATFORM] with an unauthenticated profile.	**2**
FRQ_USR_SEG_03x	**Logout**	
FRQ_USR_SEG_030	The [PLATFORM] SHALL provide a way for a user to log out of his profile (either manually or automatically). Remark: This is particularly important considering that the car could be rented.	**0**

7.2 Data protection

FRQ_USR_DAT_01x	**File system encryption**	
FRQ_USR_DAT_011	The user data SHALL be protected in confidentiality by encryption (see requirement **FRQ_FMW_CRY_010**).	**0**
FRQ_USR_DAT_012	This protection SHOULD be done by a hardware key unique per device and provisioned into IMx6 OTP area.	**1**
FRQ_USR_DAT_013	Or this key COULD be derived from the hash of the user password (see requirement **FRQ_FMW_STO_010**).	**2**
FRQ_USR_DAT_02x	**Password storage**	

FRQ_USR_DAT_021	The [PLATFORM] SHALL generate a unique salt per profile.	**0**
FRQ_USR_DAT_022	This salt SHALL be used with the user password and a supported hashing algorithms (according to **FRQ_FMW_CRY_010**) to generate the password hash.	**0**
FRQ_USR_DAT_023	The [PLATFORM] SHALL store ONLY this hash.	**0**
FRQ_USR_DAT_024	The [PLATFORM] SHALL NOT store the password itself.	**0**
FRQ_USR_DAT_03x	**Password storage in memory (RAM)**	
FRQ_USR_DAT_030	The [PLATFORM] SHALL not keep passwords in memory longer than strictly necessary.	**0**

7.3 Profile wipe

FRQ_USR_WIP_01x	**Profile wiping**	
FRQ_USR_WIP_011	A method SHALL be available to the user to remove all personal data stored in his profile (including the profile itself).	**0**
FRQ_USR_WIP_012	This method SHALL be strongly authenticated (e.g. request user confirmation and authentication when requesting profile wipe).	**0**

8 IP Connectivity, (OSI 1, 2, 3, 4): (MNO, BluEtooth (PAN), Wi-Fi, ...)

This section describes the requirements needed when connecting the [PLATFORM] to an IP network. This IP network can be 2G/3G/4G, Wi-Fi, PAN with Bluetooth or USB for example. We want to prevent connecting the [PLATFORM] to open and/or public networks to limit the attack surface and possible data leaks.
Private and/or controlled networks can be used as upper application layers take care of data protection, such as off-board authentication and data encryption.

8.1 Physical network

FRQ_CIP_PHY_01x	**Network connection**	

FRQ_CIP_PHY_010	The [PLATFORM] SHOULD NOT connect to open/public networks. It SHOULD provide unique/personal credentials to connect to the desired network: - MNO - Wi-Fi - Bluetooth	**1**
FRQ_CIP_PHY_02x	**Network authentication**	
FRQ_CIP_PHY_020	The network SHOULD authenticate the [PLATFORM].	**1**

8.2 Firewall

FRQ_CIP_FWL_01x	**Firewall**	
FRQ_CIP_FWL_010	The [PLATFORM] SHALL have a firewall to control all outgoing and incoming connections. The firewall blocks (drops) unwanted incoming and outgoing communications, while authorizing approved communications to pass through. The firewall can deal at the network layer (packet filter) or/and at the application layer (protocol filter). Example: Linux netfilter/iptables mechanism.	**0**
FRQ_CIP_FWL_02x	**Firewall rules update**	
FRQ_CIP_FWL_021	Firewall rules SHOULD be updatable.	**1**
FRQ_CIP_FWL_022	If an update is to be used, it SHALL be secured. Example: Some VPN client has the ability to update Linux netfilter rules according to policy enforced on VPN server (see Android Brand- Specific Cisco AnyConnect Packages implementation http://www.cisco.com/en/US/docs/security/vpn_client/anyconnect/anyconnect25/administration/guide/ac10_admin_mobile_android.html)	**0**
FRQ_CIP_FWL_03x	**Firewall configuration**	

FRQ_CIP_FWL_030	The [PLATFORM] firewall SHALL filter-out all network initiated communication (no TCP/UDP port open). Example: With stateful firewall deny all packets for starting communication(including ping) sent from unauthorized external devices.	**0**

8.3 Off-board endpoint (including content providers)

FRQ_CIP_OBE_01x	**Anti-Malware**	
FRQ_CIP_OBE_011	Off-board endpoint SHALL prohibit access to content provider that propagates malicious content to the [PLATFORM].	**0**
FRQ_CIP_OBE_012	A contractual commitment SHOULD be signed on this topic.	**1**
FRQ_CIP_OBE_02x	**Malware database**	
FRQ_CIP_OBE_020	Malware database content SHALL be up to date.	**0**

8.4 Permissions

FRQ_CIP_PER_01x	**Permissions**	
FRQ_CIP_PER_010	The [PLATFORM] SHALL implement a permission model to grant or refuse network connectivity depending on the authorizations of the requesting program. (see **FRQ_FMW_PER_010**)	**0**

8.5 Wireless network

FRQ_CIP_WLN_01x	**General**	
FRQ_CIP_WLN_010	The [PLATFORM] SHALL NOT use old version of the protocol. The latest version SHALL be first used (as much as possible). -Wi-Fi: WPA3 or above -Bluetooth: 2.1 or above	**0**
FRQ_CIP_WLN_02x	**Bluetooth**	

FRQ_CIP_WLN_020	The [PLATFORM] SHALL meet the following technical requirements for on-board Bluetooth communication with high security requirements, such as key exchange; a) External devices SHALL be authenticated to prevent illegal access, b) After the mobile device discovers the vehicle's Bluetooth device, it SHALL be SSP mode (for Classic occasions) or LE Secure Connection (for LE occasions). c) In order to prevent illegal access to data information, the protocol data SHALL be securely encrypted.	**0**
FRQ_CIP_WLN_021	The passkey SHALL be unique for each device.	**0**
FRQ_CIP_WLN_022	If there are no link keys or only one side has a link key, pairing SHALL NOT succeed.	**0**
FRQ_CIP_WLN_023	The [PLATFORM] SHALL NOT accept a Bluetooth packet with an out-of-spec profile/role.	**0**
FRQ_CIP_WLN_03x	**Wi-Fi**	
FRQ_CIP_WLN_030	The SSID provided by the [PLATFORM] SHOULD NOT include any information which is unique to the vehicle.	**1**
FRQ_CIP_WLN_031	The [PLATFORM] SHALL NOT automatically connect to access points which is not authorized by the user.	**0**

9 Connectivity (others: Bluetooth audio, USB mass storage, SD card, ...)

This section describes the requirements related to all connectivity except IP. We are trying to prevent unauthorized access to these systems: audios devices, USB devices, SD cards …

9.1 Permissions

FRQ_COT_PER_01x	**Permissions**	
FRQ_COT_PER_010	The [PLATFORM] SHALL implement a permission model to grant or refuse connectivity depending on the authorizations of the requesting program. (see **FRQ_FMW_PER_010**)	**0**

10 Applications

This section describes the requirements related to “applications”. Applications are user installable programs that may be distributed by a remote service know as a “store”. Applications may also be directly included in the system firmware. Applications may or may not be removable by users.

The requirements deal with the management of such applications: installation, removal, update, runtime, but also with the necessity of data isolation and protection. We need to encrypt the stored personal data to prevent data leaks, like passwords for email accounts for example.

10.1 Installation/Upgrade

FRQ_APP_INS_01x	**Installation**	
FRQ_APP_INS_010	The [PLATFORM] installer in charge of applications management (provisioning, update, remove) SHALL verify the digital signature of applications being installed on the [PLATFORM] (see **FRQ_FMW_CRY_010** for required signature algorithms)	**0**
FRQ_APP_INS_011	[PLATFORM] SHALL authenticate application store used to download new application.	**0**
FRQ_APP_INS_02x	**Upgrade**	
FRQ_APP_INS_021	The [PLATFORM] installer in charge of applications management (provisioning, update, remove) SHALL verify the digital signature of applications being upgraded on the [PLATFORM] (see **FRQ_FMW_CRY_010** for required signature algorithms)	**0**
FRQ_APP_INS_022	The [PLATFORM] installer SHOULD also verify that the public key used to sign the upgrade is the same as the original key used to provision the application during the previous installation. Example: If we have a system with several signing key dedicated to different apps provider and used for signing their applications, we should prevent a provider A from signing an upgrade for an application from provider B.	**1**

	(See Android Master key exploit).	
FRQ_APP_INS_03x	**Application signature**	
FRQ_APP_INS_030	The application signature SHOULD include the device identification number (device ID) as part the signed string. Remark: The effect of such signature algorithm will be the binding of the application to the [PLATFORM] and therefore avoid possible application transfers and (illegal) copy from a [PLATFORM] to another	**1**
FRQ_APP_INS_04x	**Automatic upgrade**	
FRQ_APP_INS_041	The [PLATFORM] SHALL provide a mechanism to check all the installed versions of applications with the remote versions available on the application store.	**0**
FRQ_APP_INS_042	This mechanism SHALL then warn the user of the upgrade availability and SHALL offer a way to upgrade. Example: • If application size is below than **x** MB, the mechanism could provide a direct download link. • If not, the [PLATFORM] could display a message to explain the upgrade procedure.	**0**

10.2 Removal

FRQ_APP_REM_01x	**Privacy protection**	
FRQ_APP_REM_010	The [PLATFORM] SHALL wipe off all of the application data at the time the application is deleted. This prevents the reuse of personal data and saturation of mass storage.	**0**
FRQ_APP_REM_02x	**Secure Application management**	
FRQ_APP_REM_021	The [PLATFORM] SHALL provide a way to remove application remotely.	**0**

FRQ_APP_REM_022	This request SHALL be performed securely: the request's source SHALL be authenticated and the request integrity SHALL be verified.	**0**

10.3 Local data protection

FRQ_APP_LOC_01x	**Application data isolation**	
FRQ_APP_LOC_010	The [PLATFORM] SHALL have an application isolation mechanism. • This mechanism allocates memory space for each application. • An application SHALL NOT be allowed to access the allocated space of another one unless specifically authorized by a permission (see **FRQ_FMW_PER_010**). It includes all application related files.	**0**
FRQ_APP_LOC_02x	**Application data confidentiality**	
FRQ_APP_LOC_021	The [PLATFORM] SHALL ensure confidentiality for all personal data stored. Example: Such data can be: • VIN, • Browser navigation data (history, passwords stored …), • Email, • Credentials used for the store, • Credentials used for web and email services, • Passwords (all the passwords), • Bluetooth profiles, • Chat, • SMS/MMS, • PIN for 3G USB device, • Payment information or billing reports	**0**
FRQ_APP_LOC_022	The [PLATFORM] SHALL encrypt all personal data (see **FRQ_FMW_CRY_010** for cryptography).	**0**
FRQ_APP_LOC_023	The encryption key SHALL NOT be stored in plain-text. It SHALL be protected (see **FRQ_FMW_STO_03x**).	**0**

	Example: • Derivation from PIN code • Stored in off-board system	
FRQ_APP_LOC_024	All Application Sensible data SHALL be immediately cleaned from memory (stack, cache, ..) after processing if they are in clear and not cyphered.	**0**

10.4 Runtime execution

FRQ_APP_RUN_01x	**Application integrity protection**	
FRQ_APP_RUN_011	For any executable code that is not part of the system firmware and authenticated by the Secure Boot (see **FRQ_SEC_SBO_010**) the [PLATFORM] SHALL provide mechanisms for authenticity and integrity checking of these codes.	**0**
FRQ_APP_RUN_012	This application integrity verification SHALL be performed before each execution of the application. Example: Android application integrity check before each execution.	**0**
FRQ_APP_RUN_02x	**Secure execution environment**	
FRQ_APP_RUN_021	The [PLATFORM] SHALL protect the execution of applications from other processes.	**0**
FRQ_APP_RUN_022	Each application SHALL run in a dedicated process (see **FRQ_FMW_PER_020** for requirement on process). Example: MMU (Memory Management Unit) on all modern processor. Kernel => direct access to physical memory Userland => access to virtual memory	**0**
FRQ_APP_RUN_023	When an application accesses user data, the platform SHALL ask the user if they allow it, and if not, SHALL deny that access.	**0**

FRQ_APP_RUN_024	The application environment SHOULD be monitored by OS in real time, and abnormal conditions (such as abnormal network connections, sudden increase in memory usage) SHOULD be alerted.	**1**
FRQ_APP_RUN_03x	**System isolation from applications**	
FRQ_APP_RUN_031	To ensure maximum security for the [PLATFORM], application SHALL NOT get direct access to low level APIs of the OS.	**0**
FRQ_APP_RUN_032	All API calls made by the applications SHALL be executed in a SandBox environment. Example: Android Dalvik VM. =>Implies that Android JNI calls are forbidden.	**0**
FRQ_APP_RUN_04x	**Misbehaving application self-detection**	
FRQ_APP_RUN_040	The [PLATFORM] SHOULD include a diagnosis application which detects and removes the applications which do not conform to their expected behavior (based on associated application security policy/access control rules).	**2**
FRQ_APP_RUN_041	It SHOULD have the ability to identify and block application software running with highly sensitive permissions (eg, ROOT permissions, permissions related to car control behavior)	**1**
FRQ_APP_RUN_05x	**Inter-applications communications**	
FRQ_APP_RUN_051	Direct communication SHALL NOT be possible between applications.	**0**
FRQ_APP_RUN_052	A service SHOULD be used to do such communication. Example: • Android intents • Android content provider	**1**
FRQ_APP_RUN_06x	**Validation process**	
FRQ_APP_RUN_060	When the application is running, the [PLATFORM] SHALL censor its improper access and operation.	**0**

10.5 Permissions

FRQ_APP_PER_01x	**Permissions**	
FRQ_APP_PER_010	The [PLATFORM] SHALL implement a permission framework for granting access to files or resources. This framework will be used to grant a file access to an application. By this method, the system can securely control who has access to a file or resource (such as the address book for instance).	**0**
FRQ_APP_PER_02x	**Permissions validation**	
FRQ_APP_PER_020	The permissions SHOULD also be customizable by letting the user choose if an application can have access or not to a given file or resource: when a newly downloaded application wants to access the address book for example, the [PLATFORM] SHOULD implement a popup to ask the user if he accepts or not the access.	**1**

10.6 Development

FRQ_APP_DEV_01x	**Secure Development Life-Cycle (SDLC)**	
FRQ_APP_DEV_010	The [PLATFORM] supplier SHALL define a Secure Development Life-Cycle and distribute this document to developers, integrators, designers, and all different actors involved in the solution development.	**0**
FRQ_APP_DEV_011	The SDLC SHALL be at the state of the art.	**0**

10.7 Off-board validation

FRQ_APP_VAL_01x	**Validation process**	
FRQ_APP_VAL_011	The application SHALL be submitted for validation.	**0**
FRQ_APP_VAL_012	The validation SHALL be done with the intent to find any incorrect operations or incorrect use of functions, like for example:	**0**

	• External network connections without the user being notified, • Access and modification of user's personal data, • Access and modification of the data on CAN bus, • Access and modification of calculator's data, • Use of the phone to make premium-rate calls, • Access the network whereas the application description says it should stay local, • Vulnerabilities, • Security requirement enforcements	
FRQ_APP_VAL_013	If an application doesn't conform to the expected behavior, it SHOULD be excluded from the store and removed from all [PLATFORM] where it is already installed.	**1**

10.8 Payment

FRQ_APP_PAY_01x	**User payment authentication and authorization**	
FRQ_APP_PAY_011	The Store SHALL authenticate the user who buys an application or something in it (subscription, apps option, …).	**0**
FRQ_APP_PAY_012	The authorization SHOULD be based on functional requirement. Example: When the customer enters his credit cards information in the Store, he could select various policies: • In-app payment : Allowed or forbidden or with limited capabilities (e.g. 50 € by day) • Application buying: Forbidden, allowed or allowed with credentials confirmation requested at final stage	**1**

10.9 SDK

SDK is a package containing code designed to help the development of software on the [PLATFORM]. It may contain standalone executables, libraries, resources, or any kind of data. It is capable of running privileged code on behalf of users and user applications. It is therefore a priority target for attackers and SHALL comply with strict security standards.

FRQ_APP_SDK_01x	**SDK limitations**	

FRQ_APP_SDK_010	The SDK SHALL NOT contain any means, methods or API that could lead to security failure. Example: System exec capabilities	**0**
FRQ_APP_SDK_02x	**SDK validations**	
FRQ_APP_SDK_020	The SDK SHALL comply to **FRQ_APP_DEV_010** and **FRQ_APP_VAL_01x**.	**0**

10.10 Connectivity

FRQ_APP_CON_01x	**Security of the connection**	
FRQ_APP_CON_010	Any application requiring IP connectivity SHALL use a secure connection through a TLS tunnel between the [PLATFORM] and the Alliance off-board platform in accordance with the FRQ_FMW_CRY_* requirements.	**0**

11 Internet browser

This section describes the requirements related to the Internet Browser. Internet Browser is a kind of application: every application requirement is also applicable to Internet Browser. As it is a privileged target for hackers, we also want to harden its configuration and requirements.
We want to disable all non-necessary features by default to limit the attack surface.

11.1 Proxy

FRQ_BRW_PRX_01x	**No direct connections to web resources**	
FRQ_BRW_PRX_011	The [PLATFORM] browser SHALL not connect directly to web resources.	**0**
FRQ_BRW_PRX_012	An off-board Web Proxy SHALL transfer all requests from [PLATFORM] browser to web resources.	**0**
FRQ_BRW_PRX_013	The application SHALL NOT link the unauthenticated network directly via browser.	**0**

11.2 Function restrictions

FRQ_BRW_FUN_01x	**Protection against malicious embedded code**	
FRQ_BRW_FUN_010	The [PLATFORM] browser SHALL offer a protection against malicious embedded code such as JavaScript, Java Applets, PDF, Flash or ActiveX for example.	**0**
FRQ_BRW_FUN_02x	**Web resources isolation**	
FRQ_BRW_FUN_020	Browser SHOULD create dedicated process for each Web resource requested, in order to permit a by process isolation (see **FRQ_FMW_PER_020** for process requirement).	**1**
FRQ_BRW_FUN_03x	**Browser's tuning**	
FRQ_BRW_FUN_031	The browser SHOULD be able to activate/deactivate functionalities.	**1**
FRQ_BRW_FUN_032	These functionalities SHOULD be disabled by default. But if the user wants to activate them, navigation thru the browser's option SHOULD offer him this possibility. Example: • JavaScript • Browser Plugins in an individual manner (including Adobe Flash, PDF, Java) • File download	**1**

11.3 Upgrade

FRQ_BRW_UPG_01x	**Browser's upgrade**	
FRQ_BRW_UPG_011	In case of the Internet Browser is not an application: The [PLATFORM] firmware releases SHALL include the browser updates.	**0**
FRQ_BRW_UPG_012	In case of the Internet Browser is an application: It SHALL comply with application requirements (**FRQ_APP_xxx_xxx**)	**0**

12 Off-board services (OSI 5, 6, 7)

This section describes the requirements applicable to off-board services. An off-board service is a system that communicates with the [PLATFORM] either by sending data or receiving data or both. Each service has to be determined by the following security criteria:

- Confidentiality (data and access)
- Data integrity
- Service continuity

Each criteria has four security levels: **Strategic** (0) / **Critical** (1) / **Standard** (2) / **Minimum** (3).

Confidentiality levels:

Only 3 levels are needed as the strategic level could not be used for such project.

- **C1:** Critical. Personal data from users: names, emails, phone numbers, credit cards, GPS history …
- **C2:** Standard. Device identification data: serial number, versions, battery state …
- **C3:** Minimum. Public data: RSS feeds, internet content, audio streaming …

Integrity levels:

Only 2 levels are needed as either data need integrity or not.

- **R1 :** Critical services for [PLATFORM]
  - HD Traffic
  - EV battery management
  - Coyote

- **R3 :** Minimal / Optional services for [PLATFORM]
  - Weather forecast
  - Paf Le chien
  - News

Continuity levels:

4 levels are present here as we could have services classify in each category.

- **T0 :** Strategic
  - Disaster recovery plan delays
    - 5 days degraded mode
    - 10 days nominal mode
  - Data loss SHALL be less than 24h of cumulated data

- **T1 :** Critic
  - Disaster recovery plan delays
    - 10 days degraded mode
    - 20 days nominal mode
  - Data loss SHALL be less than 24h of cumulated data

- **T2 :** Standard
  - Disaster recovery plan delays
    - 20 days nominal mode
  - Data loss SHALL be less than 24h of cumulated data

- **T3 :** Minimum
  - No delays are required

The service provider is in charge of delivering these off-board services. Each service provider has a level defined by the combination of Confidentiality levels (**Cx**) and Integrity levels (**Rx**) and Continuity levels (**Tx**).
The highest constraint of one of these levels SHALL be considered to define the service provider level.
Example:

- A service provider with **C1**, **R2**, **and T0** is considered as strategic (because **T0** has Strategic level).
- A service provider with **C3**, **R3**, **and T3** is considered as minimal.

Service providers can have four different levels:

- **SP0**: Service provider Strategic
- **SP1**: Critical service provider
- **SP2**: Standard service provider
- **SP3**: Minimal service provider

Information: We consider that content aggregators (like Atos) are service providers.
The requirements cover data protection when exchanging or storing information belonging to the [PLATFORM] or to the user. It also deals with the mandatory access controls and traceability related to this data.

12.1 Permissions

FRQ_OSV_PER_01x	**Permissions**	

FRQ_OSV_PER_010	The [PLATFORM] SHALL implement a permission model to grant or refuse the off-board service depending on the authorizations of the requesting program. (see **FRQ_FMW_PER_010**)	**0**
FRQ_OSV_PER_02x	**Customer approval**	
FRQ_OSV_PER_020	**C1** and **C2** data SHALL be sent only after customer approval. Example: • For **C1** this requirement could be enforced through specific pop-up message • For **C2** this requirement could be enforced through contractual clause with customer	**0**

12.2 Anonymization

FRQ_OSV_ANO_01x	**Functional requirements**	
FRQ_OSV_ANO_011	The off-board service SHALL require the very minimum set of information to enable the service.	**0**
FRQ_OSV_ANO_012	Every other piece of information SHALL be anonymized.	**0**
FRQ_OSV_ANO_02x	**Anonymization**	
FRQ_OSV_ANO_021	**C1** services SHALL anonymize every personal data not required to provide the service. Note: Once the information is anonymized, it cannot be used for its intended purpose, so the anonymization is performed off-board instead of on-board.	**0**
FRQ_OSV_ANO_022	**C2** services SHALL anonymize every personal data and every device specific data not required to provide the service.	**0**
FRQ_OSV_ANO_023	**C3** services SHALL anonymize every personal and device specific data.	**0**

FRQ_OSV_ANO_024	Service provider SHALL explain the use of each required data.	**0**

12.3 Data exchange

FRQ_OSV_EXC_01x	**Transmission**	
FRQ_OSV_EXC_010	**C1** and **C2** off-board services SHALL encrypt all data transmissions. Allowed cryptographic algorithms are detailed in **FRQ_FMW_CRY_01x**. Example: All protocols listed below SHALL use cipher suites listed in **FRQ_FMW_CRY_01x** • TLS 1.2 • TLS 1.3 • IPSec with IKE v2 • SSH v2 • VPN • Specific implementation on application level	**0**
FRQ_OSV_EXC_02x	**Data exchange information**	
FRQ_OSV_EXC_020	The [PLATFORM] SHOULD provide visual feedback when the network is used. An icon or a logo should be displayed on the central panel so the user can monitor if the widget/application currently run by the user is actually using or not the network connection.	**1**

12.4 Access control

FRQ_OSV_ACL_01x	**ACL**	
FRQ_OSV_ACL_011	**C1** off-board services SHALL authenticate all access to **C1** and **C2** data.	**0**
FRQ_OSV_ACL_012	**C2** off-board services SHALL authenticate all access to **C2** data.	**0**
FRQ_OSV_ACL_02x	**Content filtering**	

FRQ_OSV_ACL_020	Content aggregator SHALL restrict the accessible data or resources to the only ones required for the service. Example: The Internet proxy to access specific RSS feed SHALL NOT allow access to any other internet resources.	**0**
FRQ_OSV_ACL_03x	**Content aggregator proxy**	
FRQ_OSV_ACL_031	The content aggregator SHALL NOT grant direct access to the [PLATFORM]. Note: Content Aggregator collect all up to date information of available off-board services. Once it receives requests from user/device via an aggregator proxy, it asks to relevant off-board service and then responses to user/device through the proxy.	**0**
FRQ_OSV_ACL_032	This proxy service SHALL only answer to the [PLATFORM] requests.	**0**

12.5 Traceability

FRQ_OSV_TRC_01x	**Logs**	
FRQ_OSV_TRC_011	**C1** and **C2** off-board services SHALL log all accesses to **C1** and **C2** data and SHALL be able to identify who accessed **C1** or **C2** data and when.	**0**
FRQ_OSV_TRC_012	These logs are considered to have the same level of confidentiality (**Cx**) and SHALL therefore comply with following requirements. ・FRQ_OSV_ACL_011 ・FRQ_OSV_ANO_021	**0**

12.6 Security audits

FRQ_OSV_AUD_01x	**Audits**	

FRQ_OSV_AUD_010	**C1** and **C2** off-board services SHALL allow security audits to be performed on the production service to assess the effective security. Security audits will be performed on a regular basis by independent parties chosen by Renault-Nissan.	**0**

12.7 Security alerts endpoint

FRQ_OSV_AEP_01x	**Security alerts management**	
FRQ_OSV_AEP_011	**C1** off-board services SHALL provide a security endpoint to deal with security alerts. This endpoint SHALL accept or reject the alert within a maximum delay of 4 hours. A specific phone-number SHOULD be provided for the best response-time.	**0**
FRQ_OSV_AEP_012	**C2** off-board services SHALL provide a security endpoint to deal with security alerts. This endpoint SHALL accept or reject the alert within a maximum delay of 24 hours.	**0**
FRQ_OSV_AEP_013	**C3** off-board services SHALL provide a security endpoint to deal with security alerts. This endpoint SHALL accept or reject the alert within a maximum delay of 3 days.	**0**

12.8 Security policy

FRQ_OSV_SPO_01x	**Security alerts management**	
FRQ_OSV_SPO_011	**C1** and **C2** off-board services SHALL have an up to date security policy. It SHALL be able to provide such a policy upon Renault-Nissan request.	**0**
FRQ_OSV_SPO_012	**C3** off-board service SHOULD have an up to date security policy. It SHOULD be able to provide such a policy upon Renault-Nissan request.	**1**
FRQ_OSV_SPO_02x	**Security upgrade policy**	
FRQ_OSV_SPO_021	The security policy SHALL include a commitment on security upgrades delays.	**0**

FRQ_OSV_SPO_022	As soon as a public security upgrade is released, the service provider SHALL apply it within the committed delay (see **FRQ_OSV_SPO_03x** for details).	**0**
FRQ_OSV_SPO_03x	**Security upgrade delays**	
FRQ_OSV_SPO_031	Security upgrade delays are defined according to the service provider level. **SP0** Strategic service provider SHALL comply with minimal delay of **xx** hours.	**0**
FRQ_OSV_SPO_032	**SP1** Critical service provider SHALL comply with minimal delay of **xx** days.	**0**
FRQ_OSV_SPO_033	**SP2** Standard service provider SHALL comply with minimal delay of **xx** weeks.	**0**
FRQ_OSV_SPO_034	**SP3** Minimum service provider SHOULD comply with no minimal delay.	**0**
FRQ_OSV_SPO_04x	**Security dashboards**	
FRQ_OSV_SPO_041	**SP0** and **SP1** services providers SHALL provide security dashboards. These dashboards SHALL contain information and figures on security related events.	**0**
FRQ_OSV_SPO_042	**SP2** services providers SHOULD provide security dashboards. These dashboards SHOULD contain information and figures on security related events.	**1**

12.9 Risk analysis

FRQ_OSV_RAN_01x	**Risk analysis**	
FRQ_OSV_RAN_011	**C1** and **C2** off-board services SHALL have an up to date risk analysis on the provided service. It SHALL be able to provide such an analysis upon Renault-Nissan request.	**0**

FRA_OSV_RAN_012	**C3** off-board service SHOULD have an up to date risk analysis on the provided service. It SHALL be able to provide such an analysis upon Renault-Nissan request.	**1**

12.10 Authentication

FRQ_OSV_AUT_01x	**User and/or device identifier(s)**	
FRQ_OSV_AUT_011	For **C1** and **C2** services the service provider SHALL provide a mean for user and/or device identification with its off-board infrastructure.	**0**
FRQ_OVS_AUT_012	For **C3** services the service provider SHOULD provide a mean for user and/or device identification with its off-board infrastructure. The mean is to uniquely identify a [PLATFORM] or a user. This could be done by using the device serial number or the profile currently running on the [PLATFORM]. The off-board service provider will then be able to keep track of all the actions performed by the user on a given [PLATFORM] and/or for a given profile.	**1**
FRQ_OSV_AUT_02x	**User and/or device authentication**	
FRQ_OSV_AUT_021	For **C1** and **C2** services the service provider SHALL authenticate the user for sensitive operations (downloads, profile management, password changes, credit card addition/removal …). For this kind of communication between the [PLATFORM] and the server, the latter SHALL authenticate the user, for instance when buying applications. This can be done by the use of a PIN or password.	**0**
FRQ_OSV_AUT_022	These credentials SHALL be stored according to **FRQ_FMW_STO_010**.	**0**
FRQ_OSV_AUT_03x	**Service provider authentication**	
FRQ_OSV_AUT_031	The [PLATFORM] SHALL authenticate **R1** service providers and all received data.	**0**

FRQ_OSV_AUT_032	The [PLATFORM] SHOULD authenticate **R3** service providers and all received data.	**1**
FRQ_OSV_AUT_033	These authentications SHALL be performed according to **FRQ_FMW_CRY_010**.	**0**

12.11 Data storage

FRQ_OSV_DAS_01x	**Confidentiality**	
FRQ_OSV_DAS_010	**C1** and **C2** services providers SHALL encrypt all stored data. Encryption algorithms SHALL comply with **FRQ_FMW_CRY_010**.	**0**
FRQ_OSV_DAS_02x	**Data retention**	
FRQ_OSV_DAS_020	**C1** and **C2** services providers SHALL provide a data retention policy, compliant with legal requirements and for maximum 1 year.	**0**
FRQ_OSV_DAS_03x	**Datacenter location**	
FRQ_OSV_DAS_030	Service providers SHALL locate and commit on the geographical location of his data storage.	**0**

12.12 Service Continuity

FRQ_OSV_SCO_01x	**Continuity levels definition**	
FRQ_OSV_SCO_011	For **T0** and **T1** levels the service provider SHALL define a Business Continuity Plan (BCP). This BCP describes the organization in charge of managing service continuity during: • Disaster • Restart in degraded mode • Restart in nominal mode	**0**
FRQ_OSV_SCO_012	The **T0** and **T1** BCP SHALL include: • training of involved actors • simulated once by year	**0**

	• updated once a year according to simulation and training results	
FRQ_OSV_SCO_013	For **T2** level the service provider SHOULD define a Business Continuity Plan (BCP). This BCP describes the organization in charge of managing service continuity during: • Disaster • Restart in degraded mode • Restart in nominal mode	**1**

13 CAN controller

This section describes the requirements covering CAN-Ext and CAN-M access. We are trying to prevent malicious applications from tampering with the infotainment system.

We want to prevent malicious applications and programs from tampering with the car ECUs for which the [PLATFORM] have functional interaction, by controlling and limiting the syntax and the format of these legitimate functional interaction.

We want to prevent the spoofing of CAN frames that could lead to exploit of other ECUs present on the CAN-Ext or CAN-M.

We finally want to prevent CAN-Ext or CAN-M overloading (and ultimately CAN BUS-OFF) by limiting the writing rate.

For information, other ECUs are not reachable by the [PLATFORM] (filtered-out by a dedicated component).

13.1 Separate controller

FRQ_CAN_CTR_01x	**Physical function segregation**	
FRQ_CAN_CTR_010	The controller in charge of the CAN connectivity (reading from and writing to CAN buses) SHOULD be a standalone controller (physically separate from the main controller). The [PLATFORM] CAN controller acts as a server to other components like the main controller. This enables to enforce a security policy on the [PLATFORM] CAN controller directly.	**1**
FRQ_CAN_CTR_012	Physical separation MAY be enforced through virtualization if the hypervisor is proven to be at least as secure as the physical separation:	**2**

	• hardware assisted virtualization, • certifications : common criteria (EAL4+,…), FIPS, … • …	
FRQ_CAN_CTR_013	The demonstration SHALL be made by the supplier.	**0**
FRQ_CAN_CTR_02x	**Firmware**	
FRQ_CAN_CTR_020	[PLATFORM] CAN controller SHALL comply with: • **FRQ_FMW_UPD_xxx** • **FRQ_FMW_CRY_010** • **FRQ_FMW_DAT_xxx** • **FRQ_FMW_STO_03x**	**0**
FRQ_CAN_CTR_03x	**Functions hosting**	
FRQ_CAN_CTR_030	The [PLATFORM] CAN controller SHALL only host functions related to CAN environment. It is strictly forbidden to use [PLATFORM] CAN controller resources for any other purpose.	**0**

13.2 Command filtering

FRQ_CAV_FIL_01x	**CAN Controller design**	
FRQ_CAV_FIL_011	The CAN controller SHALL filter the commands written to the CAN.	**0**
FRQ_CAV_FIL_012	The controller SHALL implement the syntax of CAN commands and only accept range-defined parameters from clients.	**0**
FRQ_CAV_FIL_013	The request made to the CAN controller SHALL be in the general form: - "I want to issue a request of type '*read*' to ECU '*air conditioning*' with the first parameter being '*temperature'* - "I want to issue a request of type '*write*' to ECU '*air conditioning*' with the first parameter being '*temperature'* and the second being '*22*'.	**0**

FRQ_CAN_FIL_01x	**Filtering**	
FRQ_CAN_FIL_011	The [PLATFORM] CAN controller SHALL implement a frequency threshold protection for each CAN command. Example: In the case that the frequency is 100ms. Packets can be treated as follows. 1. If the frequency of received CAN command is less than 10ms, it should be dropped. 2. Otherwise, it should be accepted.	**0**
FRQ_CAN_FIL_012	It SHALL drop any subsequent requests above the threshold.	**0**
FRQ_CAN_FIL_013	This threshold SHALL be configurable per command.	**0**

13.3 Controller upgrade

FRQ_CAN_UPG_01x	**Controller upgrade**	
FRQ_CAN_UPG_011	The [PLATFORM] CAN controller SHALL be upgradable through at least one the available interfaces (**OEM, USER** or **REMOTE)**.	**0**
FRQ_CAN_UPG_012	The CAN controller SHALL verify and authenticate the upgrade image coming from **any** interfaces (according to **FRQ_FMW_UPD_xxx**).	**0**

14 Development / Serial Life

This section describes the requirements managing definitions and contents of Secure and Unsecure state for parts. These distinctions are done to differentiate parts during development and serial life.

The objective is to manage properly all debug features needed during part development and the necessity to limit/control their uses on parts delivered to end customer.
For this purpose, we have to consider two hardware definitions: **Development part** and **Serial Life part** and two firmware definitions: **Secure firmware** and **Unsecure firmware**.

Consequently, we have to manage the mix between hardware and firmware definitions (2^2 combinations).

FRQ_DSL_MIX_01x	**Authorized combination**	
FRQ_DSL_MIX_011	Hardware **Development part** COULD run **Secure firmware** and **Unsecure firmware.**	**2**
FRQ_DSL_MIX_012	Hardware **Serial Life part** SHOULD run ONLY **Secure firmware**.	**1**
FRQ_DSL_MIX_013	Hardware **Serial Life part** SHOULD NOT be able to run **Unsecure firmware**.	**1**
FRQ_DSL_MIX_02x	**Special conditions for Serial Life part**	
FRQ_DSL_MIX_020	If **FRQ_DSL_MIX_012** and **FRQ_DSL_MIX_013** are not met, then **FRQ_DSL_SEL_xxx** SHALL be met.	**0**
FRQ_DSL_SEL_01x	**Secure and Unsecure States differentiations**	
FRQ_DSL_SEL_010	Based on authorized combinations described in **FRQ_DSL_MIX_01x** the parts SHOULD have a dedicated hardware. Development part HW SHOULD NOT directly produced as Serial Life part HW.	**1**
FRQ_DSL_SEL_02x	**Production of unsecure parts**	
FRQ_DSL_SEL_021	Development parts and Serial Life parts SHALL have a different product reference to avoid assembly errors.	**0**
FRQ_DSL_SEL_022	Supplier SHALL provide to Renault-Nissan a set of development part samples before launching production of Serial Life parts.	**0**
FRQ_DSL_SEL_023	Supplier SHALL be able to provide to Renault-Nissan new development parts all along Serial Life part production. This implies that the supplier is able to handle several product references at all times.	**0**
FRQ_DSL_SEL_03x	**Secure and Unsecure states**	
FRQ_DSL_SEL_030	Development parts and Serial Life parts SHALL use different cryptographic material (see FRQ_FMW_CRY_010 for cryptographic algorithms authorized) and specific process to	**0**

	authorize or not installation of Unsecure firmware or Secure firmware.	

15.1 Hardware definitions

In this chapter we will describe what it is possible to do on Development parts and what is forbidden on Serial Life parts.

FRQ_DSL_HWR_01x	**Development part hardware definition**	
FRQ_DSL_HWR_011	Development parts SHALL have a dedicated hardware public key used for secure boot. This hardware key SHALL be different between Renault-Nissan and between Development / Serial Life parts.	**0**
FRQ_DSL_HWR_012	During [PLATFORM] development, the parts MAY host debug ports like physical JTAG port, physical UART connector on the board. If development tools and equipment are used they SHALL be shared with Alliance	**2**
FRQ_DSL_HWR_02x	**Serial life part hardware definition**	
FRQ_DSL_HWR_021	For serial life, the parts SHALL NOT host debug ports (like physical JTAG port, physical UART connector) on the board.	**0**
FRQ_DSL_HWR_022	Corresponding pin pad on the board SHOULD NOT be present.	**1**
FRQ_DSL_HWR_023	This part SHALL be the only kind of hardware used by end customer.	**0**
FRQ_DSL_HWR_024	Serial life parts SHALL have a dedicated hardware public key used for secure boot. This hardware key SHALL be different between Renault-Nissan and between Development / Serial Life parts.	**0**
FRQ_DSL_HWR_025	To resist against reverse engineering the [PLATFORM] SHOULD reduce physical attack surface on HW such as; -exposed pins -readable silkscreen with marked ports or pins	**1**

15.2 Firmware definitions

In this chapter we will describe what are the contents and the purpose of the **Secure firmware** and the **Unsecure firmware.**

FRQ_DSL_SWR_01x	**Unsecure firmware**	
FRQ_DSL_SWR_011	During [PLATFORM] development, the firmware MAY include all relevant tools or debug functionalities. Example: • ADB binary • BusyBox • Root shell • Serial port output/input	**2**
FRQ_DSL_SWR_012	The **Unsecure firmware** SHALL be authenticated by Secure Boot (see **FRQ_FMW_SBO_xxx**).	**0**
FRQ_DSL_SWR_013	This verification SHALL be done by a Public Key dedicated to **Unsecure firmware** verification (see **FRQ_FMW_SBO_020**).	**0**
FRQ_DSL_SWR_014	The associated Private Key SHALL be used only for this purpose.	**0**
FRQ_DSL_SWR_02x	**Secure firmware**	
FRQ_DSL_SWR_020	**Secure firmware** SHALL fulfill all requirements related to firmware (see **FRQ_FMW_xxx_xxx**).	**0**
FRQ_DSL_SWR_021	The secure firmware SHALL not include any debug symbol that may allow a debugging tool to access to information from the source code (ex: name of the identifiers).	**0**

15.3 Unlocked mode on Serial Life parts

Serial Life parts may need to be diagnosed through unsecure features (serial port, JTAG, …) that are not allowed on nominal state with Serial life parts. A robust mechanism has to be implemented to enable these unsecure features on serial life parts.

FRQ_DSL_UNS_01x	**Challenge generation**	

FRQ_DSL_UNS_011	The [PLATFORM] SHALL issue a random challenge upon an unlocking request. The Challenge SHALL be based on random part and Serial ID of the [PLATFORM]	**0**
FRQ_DSL_UNS_012	The random part of the challenge SHALL be issued from a robust entropy source Ref : NIST SP 800-90A NIST SP 800-90B.	**0**
FRQ_DSL_UNS_013	The random part of the challenge SHALL be at least 128 bits long.	**0**
FRQ_DSL_UNS_014	The random part SHOULD be persistent. Response could be provided after [PLATFORM] power off.	**1**
FRQ_DSL_UNS_015	Challenge SHALL be extract from the [PLATFROM] thanks to specific routine control and then provided to Alliance team for Response generation.	**0**
FRQ_DSL_UNS_016	If the random part is saved in a persistent memory, it SHALL be erased after unlock success.	**0**
FRQ_DSL_UNS_02x	**Response Generation**	
FRQ_DSL_UNS_020	Signature SHALL be made by using an RSA Signature algorithm.	**0**
FRQ_DSL_UNS_021	[PLATFORM] SHALL support signature with 2048 and 3072 bits length.	**0**
FRQ_DSL_UNS_022	[PLATFORM] SHALL support signature scheme PKCS#1 V2.2 RSASSA-PSS (RFC 8087).	**0**
FRQ_DSL_UNS_023	Response generation signature SHALL use a salt length defined to 32 and SHA 256 as hash algorithm. Example of equivalent openssl command : openssl dgst -sha256 -sigopt rsa_padding_mode:pss -sigopt rsa_pss_saltlen:32 -sign privKey.key -hex -out signature.bin	**0**
FRQ_DSL_UNS_024	Response SHALL use hexadecimal format.	**0**
FRQ_DSL_UNS_03x	**Response Validation**	
FRQ_DSL_UNS_030	[PLATFORM] SHALL switch to unlocked mode only after successful response validation.	**0**
FRQ_DSL_UNS_031	Response SHALL be injected in [PLATFORM] thanks to specific DID command	**0**
FRQ_DSL_UNS_032	[PLATFORM] SHALL validate the response by checking ·Signature thanks to embedded RSA public key ·Random part is same than the random embedded in the challenge	**0**

	·Serial ID is the Unique ID of the [PLATFORM] Example of equivalent openssl command for signature check : openssl. dgst -sha256 -sigopt rsa_padding_mode:pss -sigopt rsa_pss_saltlen:32 -signature signature.bin -verify pubKey.key Challenge	
FRQ_DSL_UNS_033	RSA public key SHALL only be used for this lock/unlock purpose.	**0**
FRQ_DSL_UNS_034	Process in charge of checking response SHALL be implemented on part component that will have the highest security level if lock / unlock feature is used for other things than logs on this component. RSA public key used for response validation SHALL be stored on the component where validation process is executed Ex : In a ECU with a SOC, a TEE and a MICOM we consider MICOM as the component with the highest security level. If lock/unlock feature is only used for logs on the MICOM the response validation process will be implemented in TEE. If lock/unlock feature offer other privileges on MICOM like memory dump, code execution, debugging feature then response validation process shall be implemented in MICOM * The part component with “Highest security level” shall be determined at the stage of RFQ or specification creation in consideration of ECU architecture.	**0**
FRQ_DSL_UNS_04x	**Unlocking Certificate Authority**	
FRQ_DSL_UNS_040	The [PLATFORM] MAY store a dedicated certificate authority to authenticate the challenge request signature.	**2**
FRQ_DSL_UNS_05x	**Anti-brute force**	
FRQ_DSL_UNS_050	The [PLATFORM] SHALL implement an Anti-brute force mechanism for this mode selection.	**0**
FRQ_DSL_UNS_051	The Anti-Brute force mechanism SHALL be robust against part reboot Example:	**0**

	Upon first failure, the next challenge is issued 5 seconds later. Upon second failure, the next challenge is issued 1 minute later. Upon third failure, the next challenge is issued 1 hour later.	
FRQ_DSL_UNS_06x	**Reverting**	
FRQ_DSL_UNS_061	The [PLATFORM] SHALL not implement a mechanism to revert an unsecure part to a secure one.	**0**
FRQ_DSL_UNS_062	The only way to recover a secure serial life part SHALL be to wipe completely the configuration and reflash a serial life firmware	**0**
FRQ_DSL_UNS_07x	**Unlocked mode**	
FRQ_DSL_UNS_070	The secure firmware SHALL only include the necessary tools to provide troubleshooting functions when switched to unlocked mode through the described mechanism.	**0**
FRQ_DSL_UNS_071	The unlocked mode SHALL not be persistent. Example: ・ECUs SHALL change the mode from unlock mode to locked mode after reboot. ・If there is no operation for a certain period, ECUs SHALL change from unlock mode to locked mode.	**0**
FRQ_DSL_UNS_072	[PLATFORM] SHALL return automatically to regular state after a switch off.	**0**
FRQ_DSL_UNS_073	The unlocked mode activation SHALL follow Alliance unlocking protocol [REF : C1A HS– Gateway Security functional requirement]	**0**
FRQ_DSL_UNS_074	The secure firmware SHALL not include any debug symbol that may allow a debugging tool to access to information from the source code (ex: name of the identifiers).	**0**